4

DBMS - Mini Project NGO Course Management System

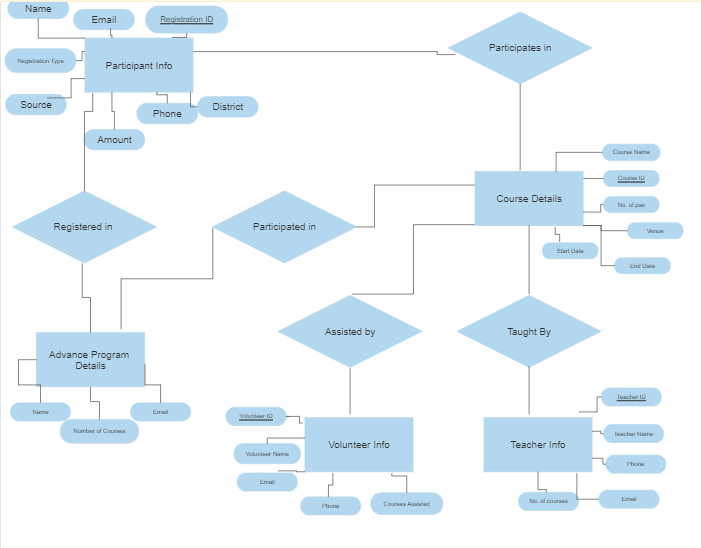
Submitted by:

Kshitij G Saha

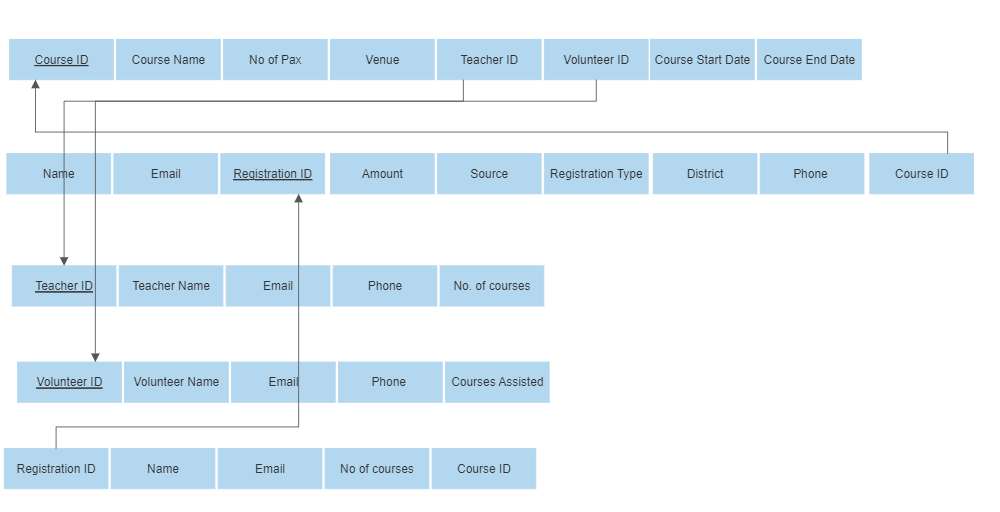
PES1UG20CS222

Section D

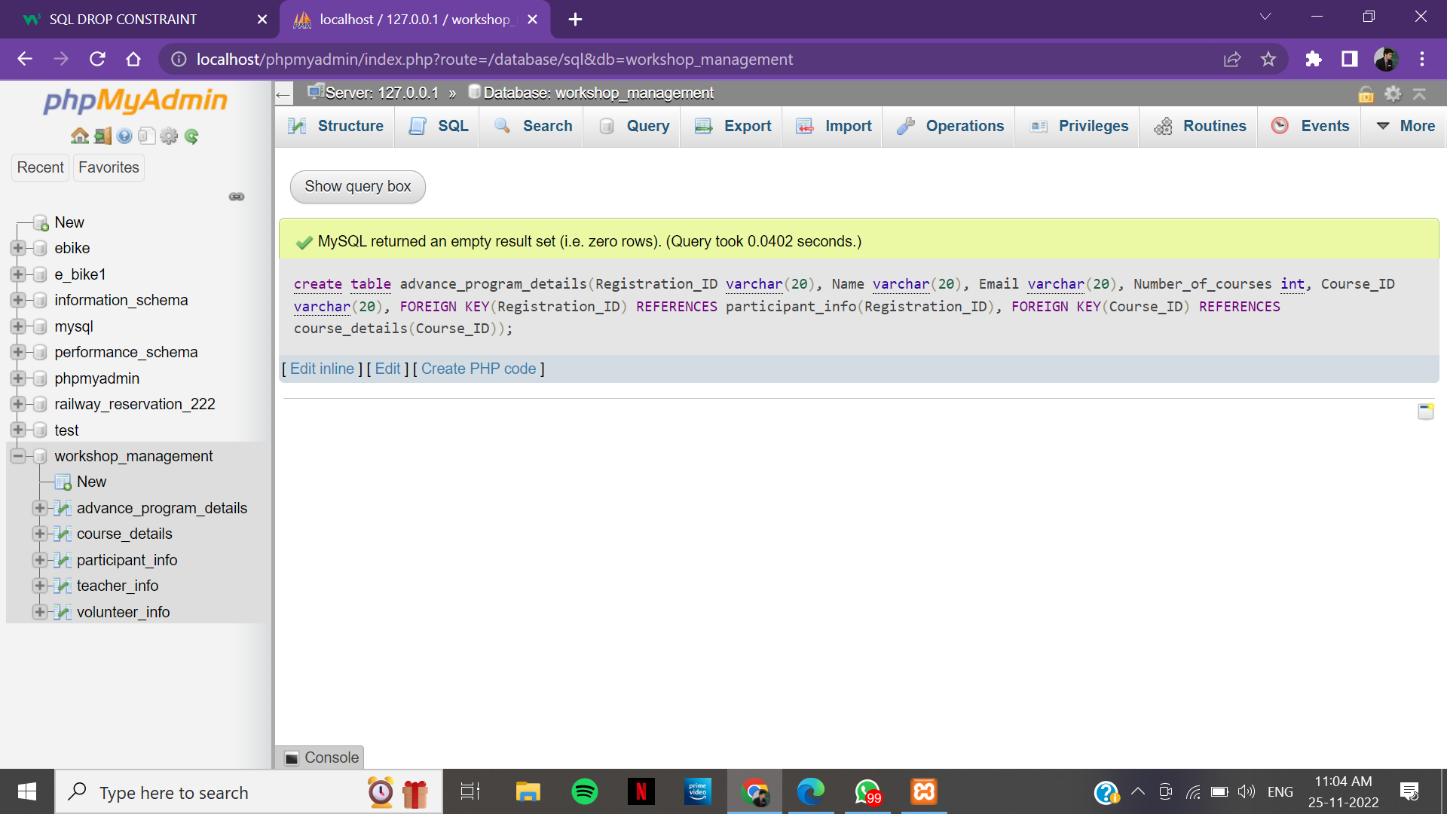
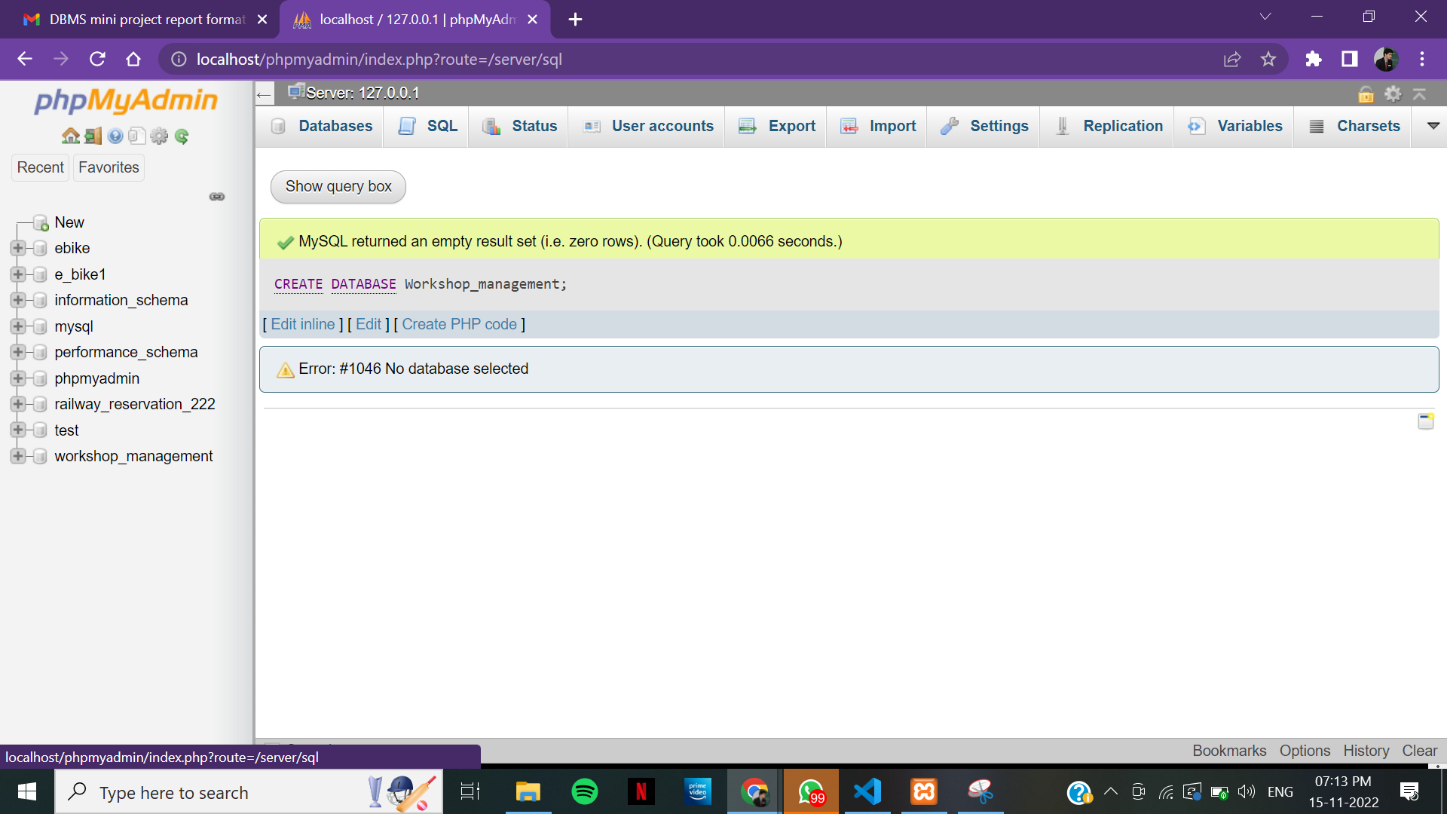
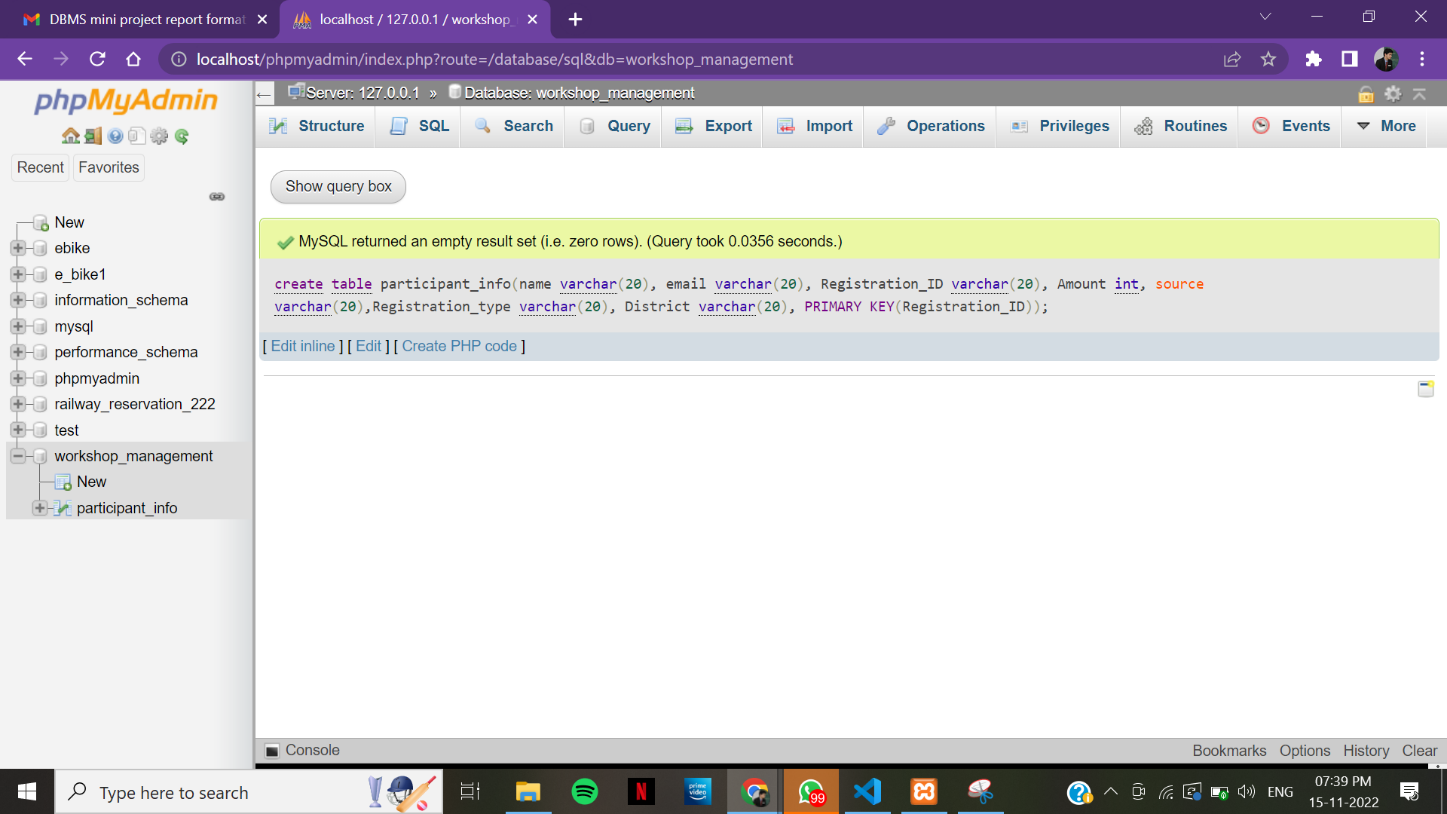
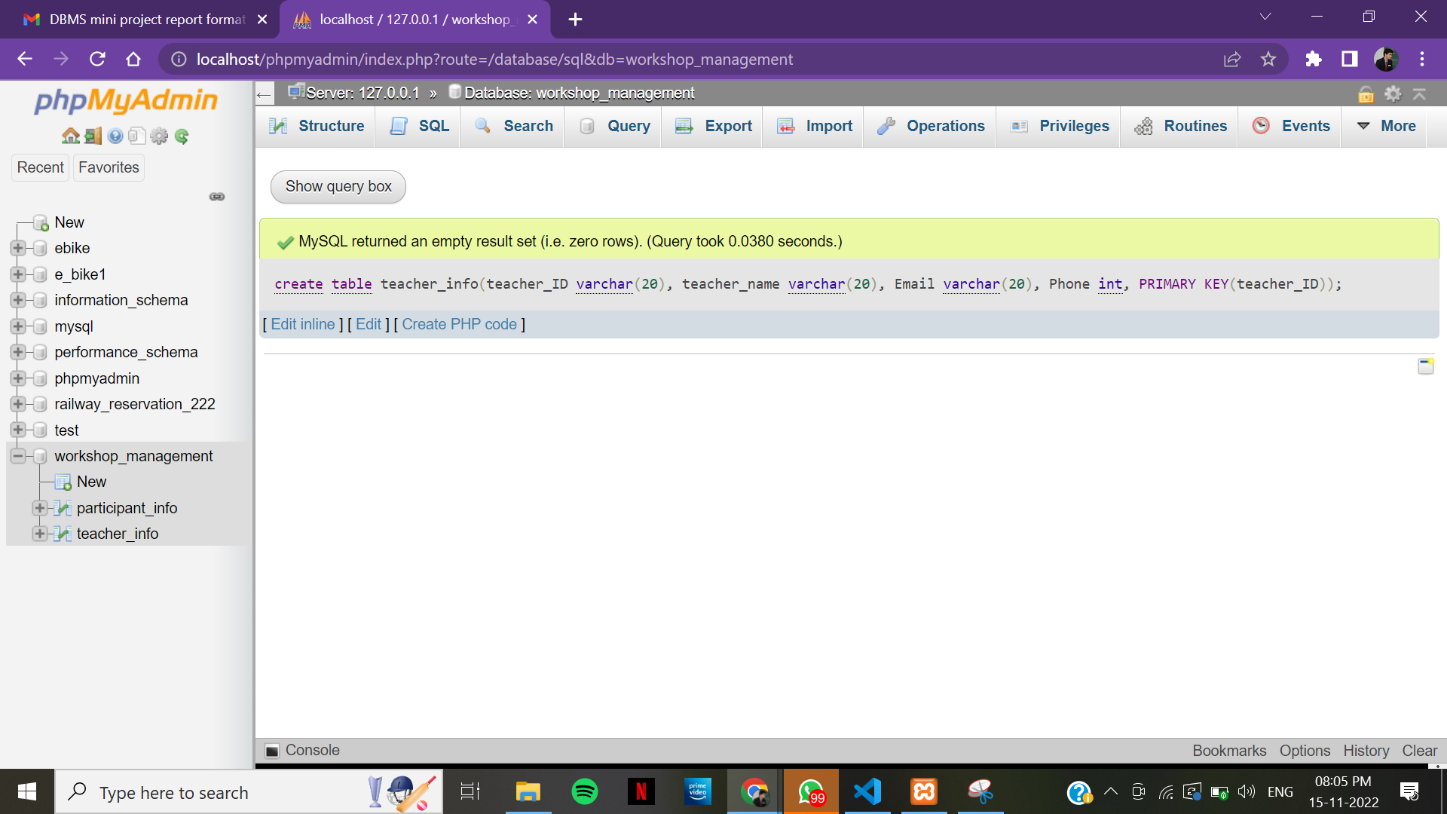
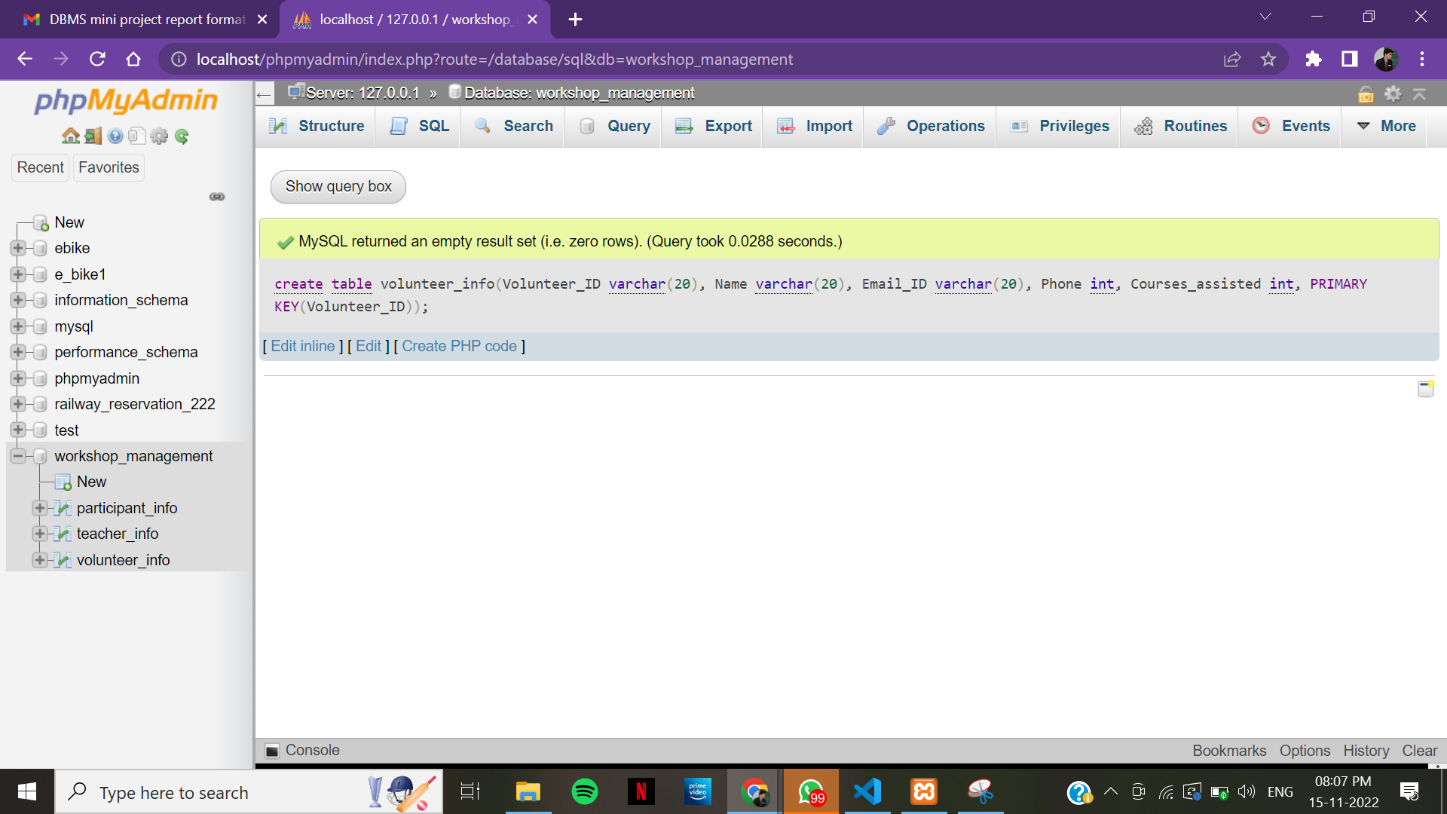
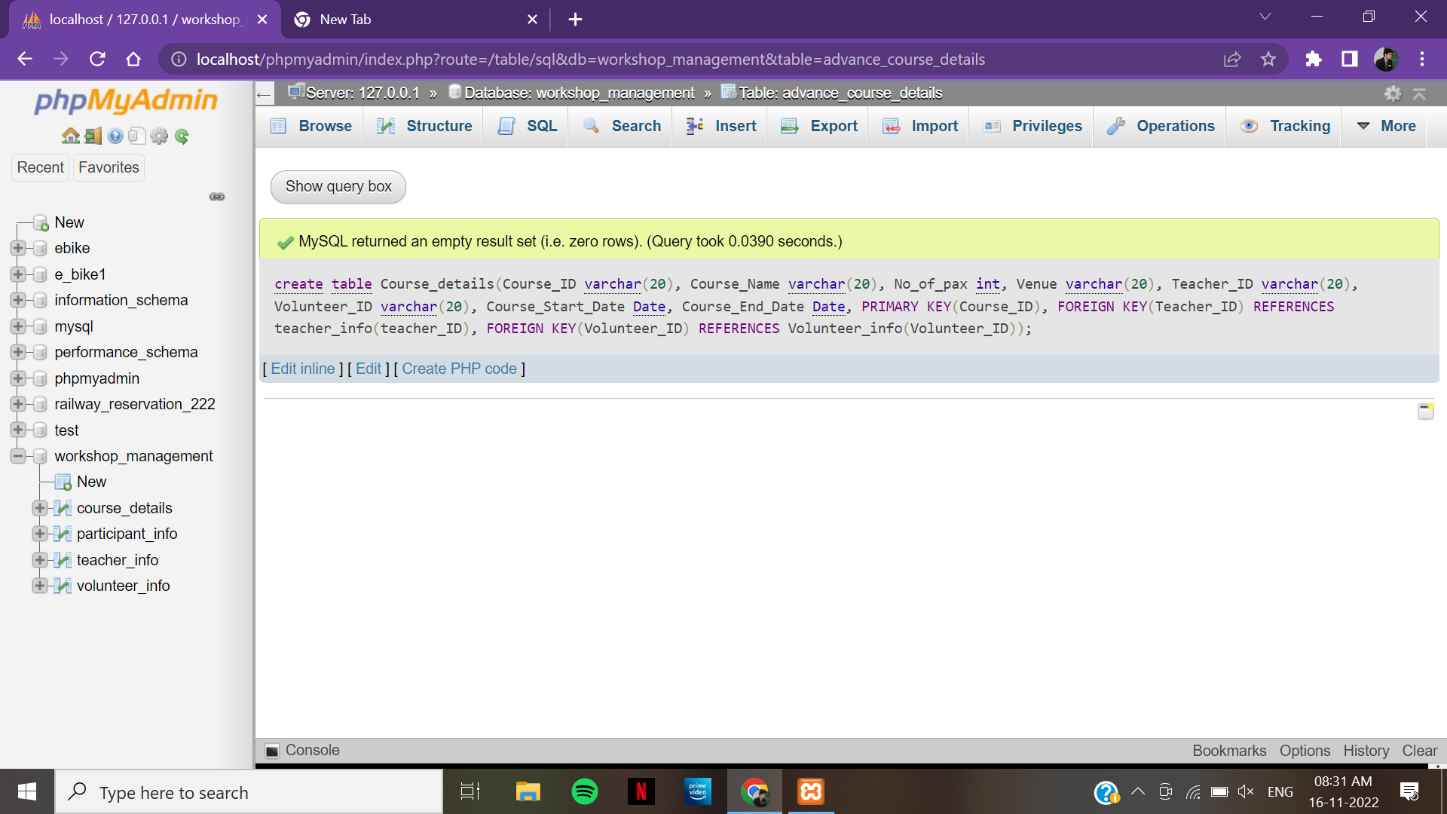
**ER Diagram**



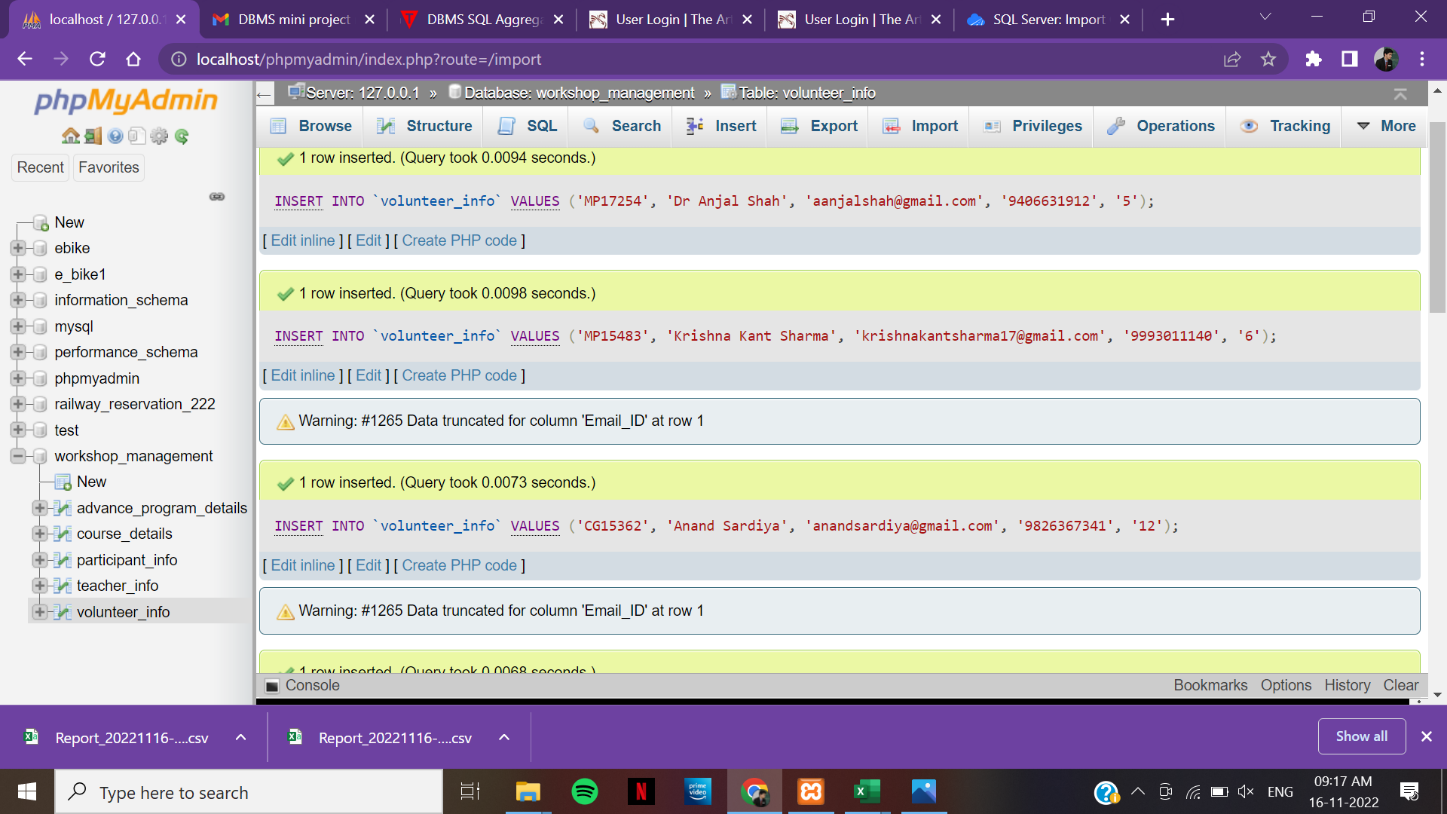
# Relational Schema



**DDL statements - Building the database**



# Populating the Database



**Join Queries**

Showcase at least 4 join queries

Write the query in English Language, Show the equivalent SQL statement and also a screenshot of the query and the results

/\* JOIN course\_details and teacher\_info with teacher\_ID\*/

SELECT \* FROM course\_details INNER JOIN teacher\_info ON course\_details.teacher\_ID = teacher\_info.teacher\_ID;

/\* JOIN course\_details and volunteer\_info with volunteer\_ID\*/

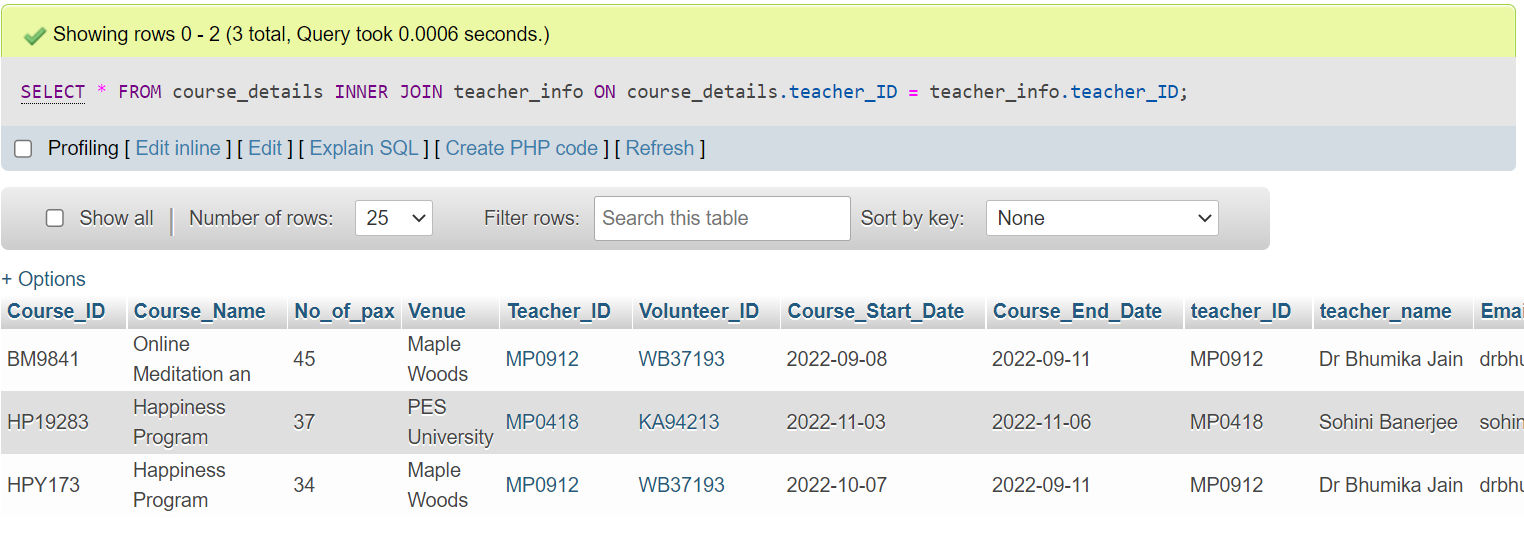
SELECT \* FROM course\_details INNER JOIN volunteer\_info ON course\_details.Volunteer\_ID = volunteer\_info.volunteer\_ID;

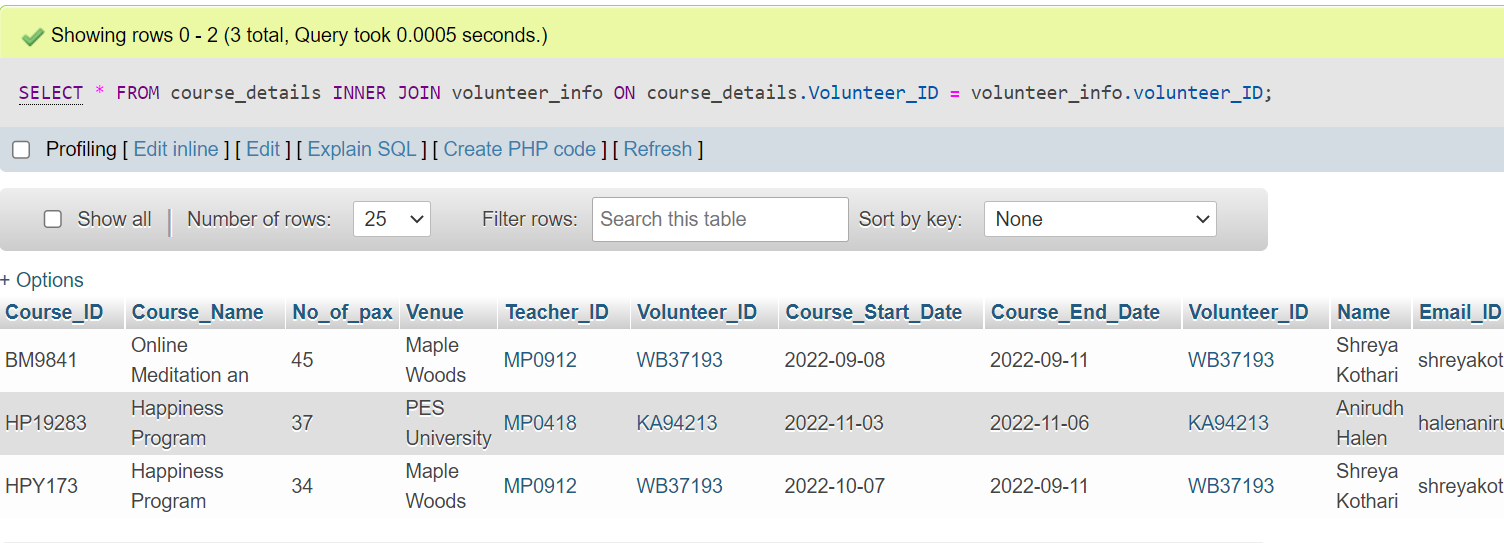
/\* JOIN teacher\_info and course\_details with teacher\_ID\*/

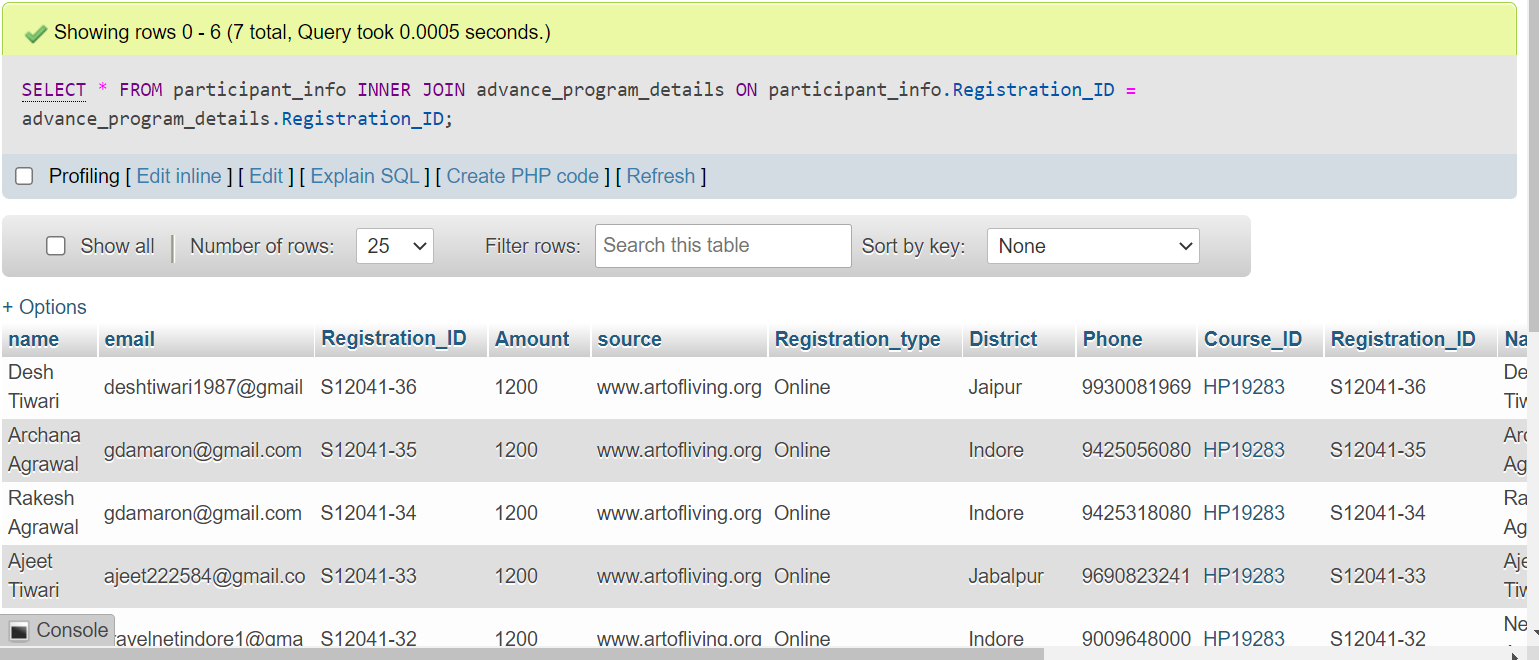
SELECT \* FROM teacher\_info INNER JOIN course\_details ON teacher\_info.teacher\_ID = course\_details.teacher\_ID;

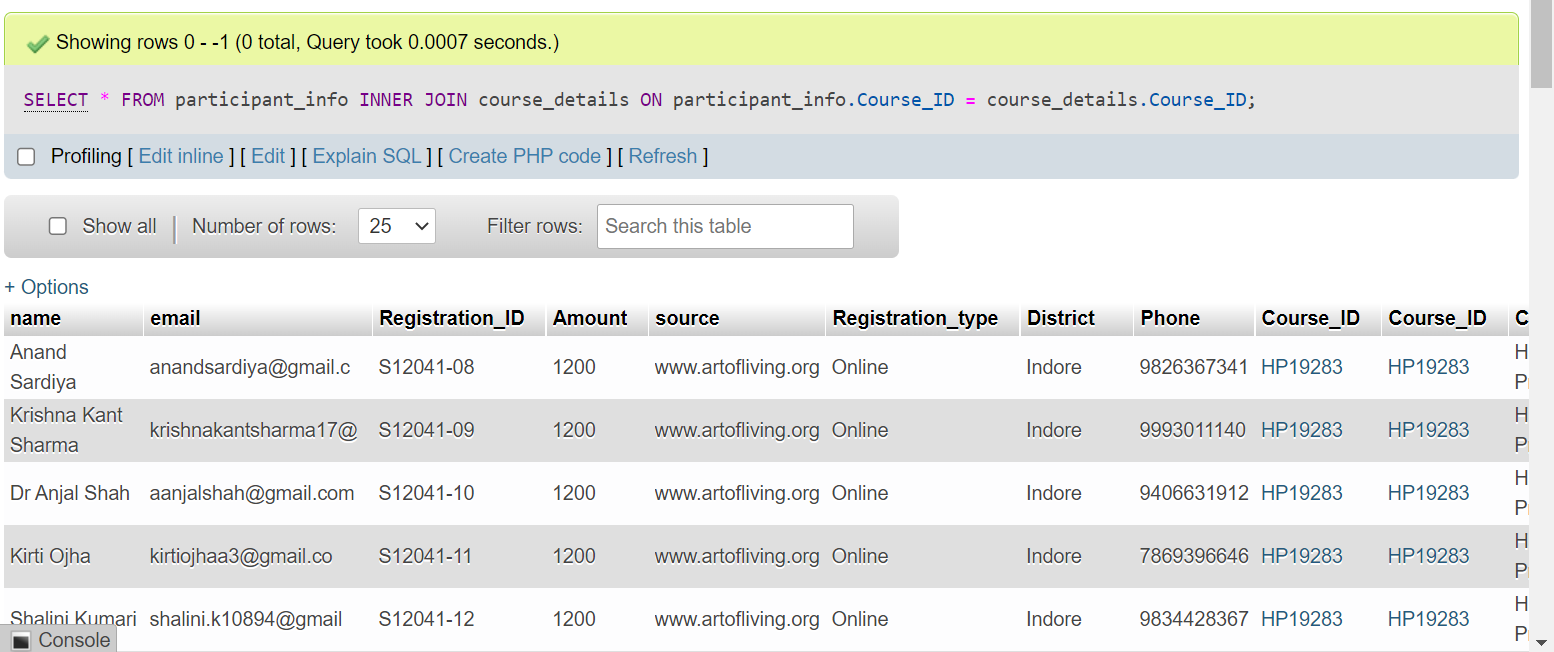
/\* Join participant\_info and course\_details with Course\_ID\*/

SELECT \* FROM participant\_info INNER JOIN course\_details ON participant\_info.Course\_ID = course\_details.Course\_ID;









# Aggregate Functions

Showcase at least 4 Aggregate function queries

Write the query in English Language, Show the equivalent SQL statement and also a screenshot of the query and the results

/\*Aggregate function to select teacher with max no\_of-courses\*/

SELECT \* FROM teacher\_info WHERE no\_of\_courses = (SELECT MAX(no\_of\_courses) FROM teacher\_info);

/\*Aggregate function to select volunteer with max Courses\_assisted\*/

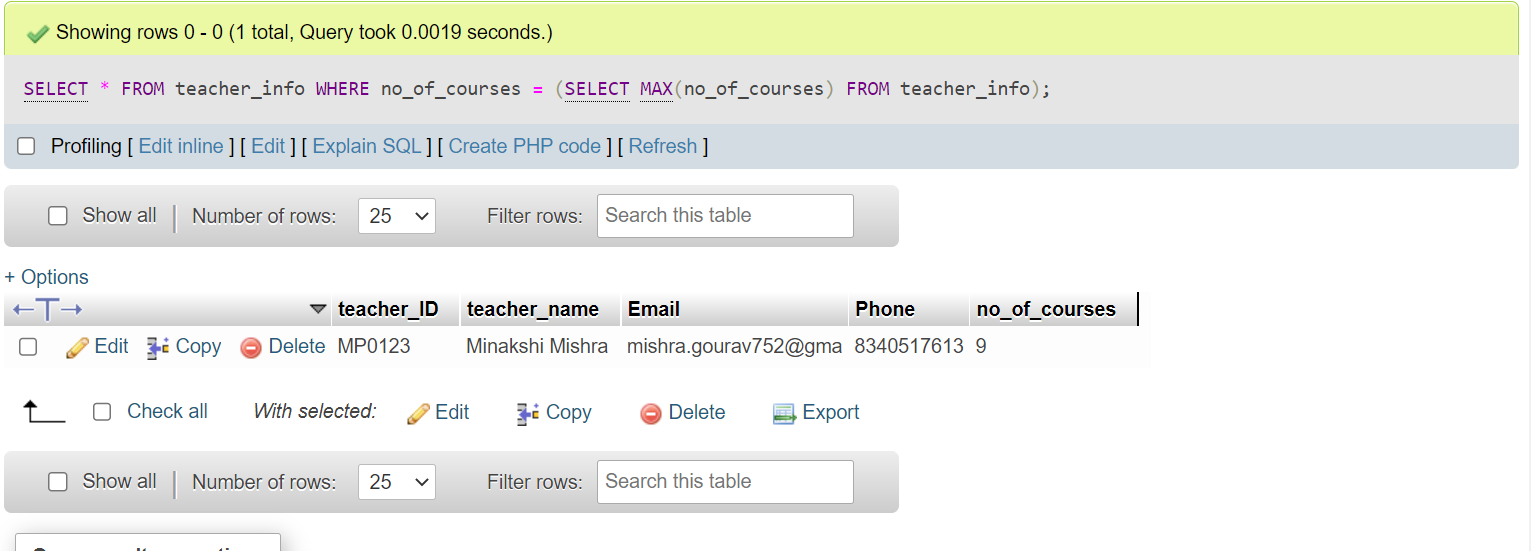
SELECT \* FROM volunteer\_info WHERE Courses\_assisted = (SELECT MAX(Courses\_assisted) FROM volunteer\_info);

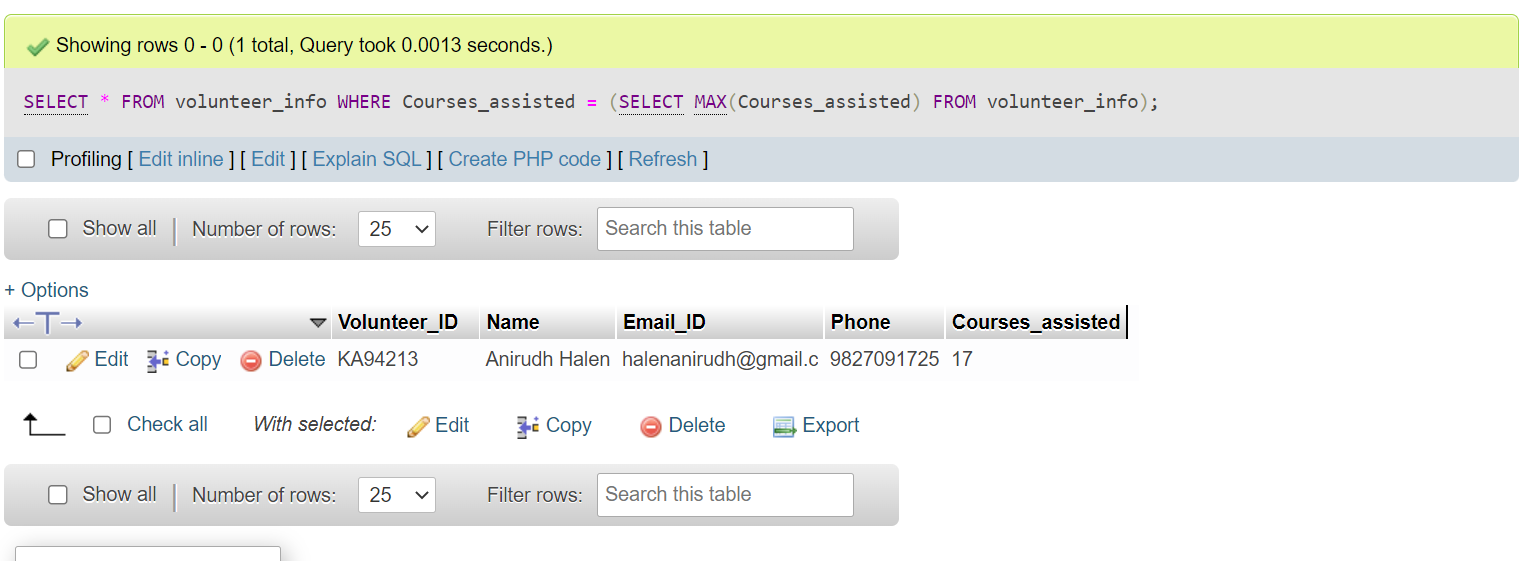
/\*Aggregate function to Show total number of courses in course\_details table\*/

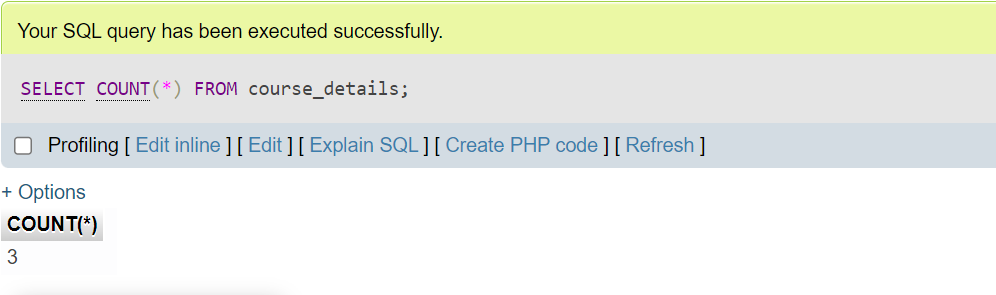
SELECT COUNT(\*) FROM course\_details;

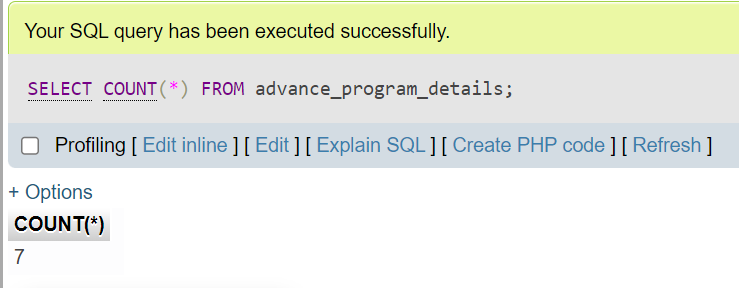
/\*Aggregate function to show total number of participants in advance\_program\_details table\*/

SELECT COUNT(\*) FROM advance\_program\_details;









# Set Operations

Showcase at least 4 Set Operations queries

Write the query in English Language, Show the equivalent SQL statement and also a screenshot of the query and the results

/\* Set operation to get list of course\_details with Course\_Start\_Date as 2022-09-08 and Course\_End\_Date as 2022-09-11 and No\_of\_pax is above 40\*/

SELECT \* FROM course\_details WHERE Course\_Start\_Date = '2022-09-08' AND Course\_End\_Date = '2022-09-11' AND No\_of\_pax > 40;

/\*Set operation to get list of participants where Amount is 1200 and Registration\_type is 'Online'\*/

SELECT \* FROM participant\_info WHERE Amount = 1200 AND Registration\_type = 'Online';

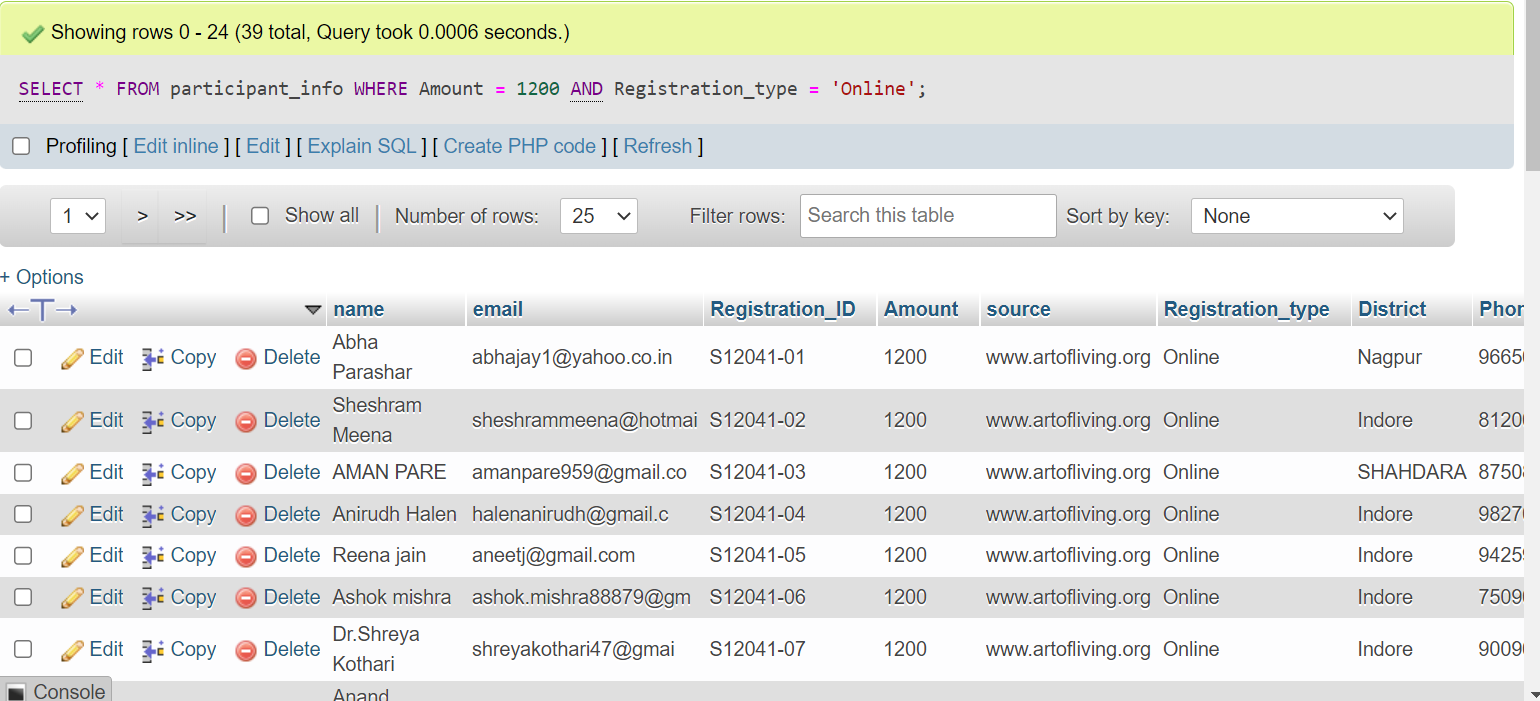
/\*Set operation to get list of participants whose District is Indore or Bhopal or Damoh\*/

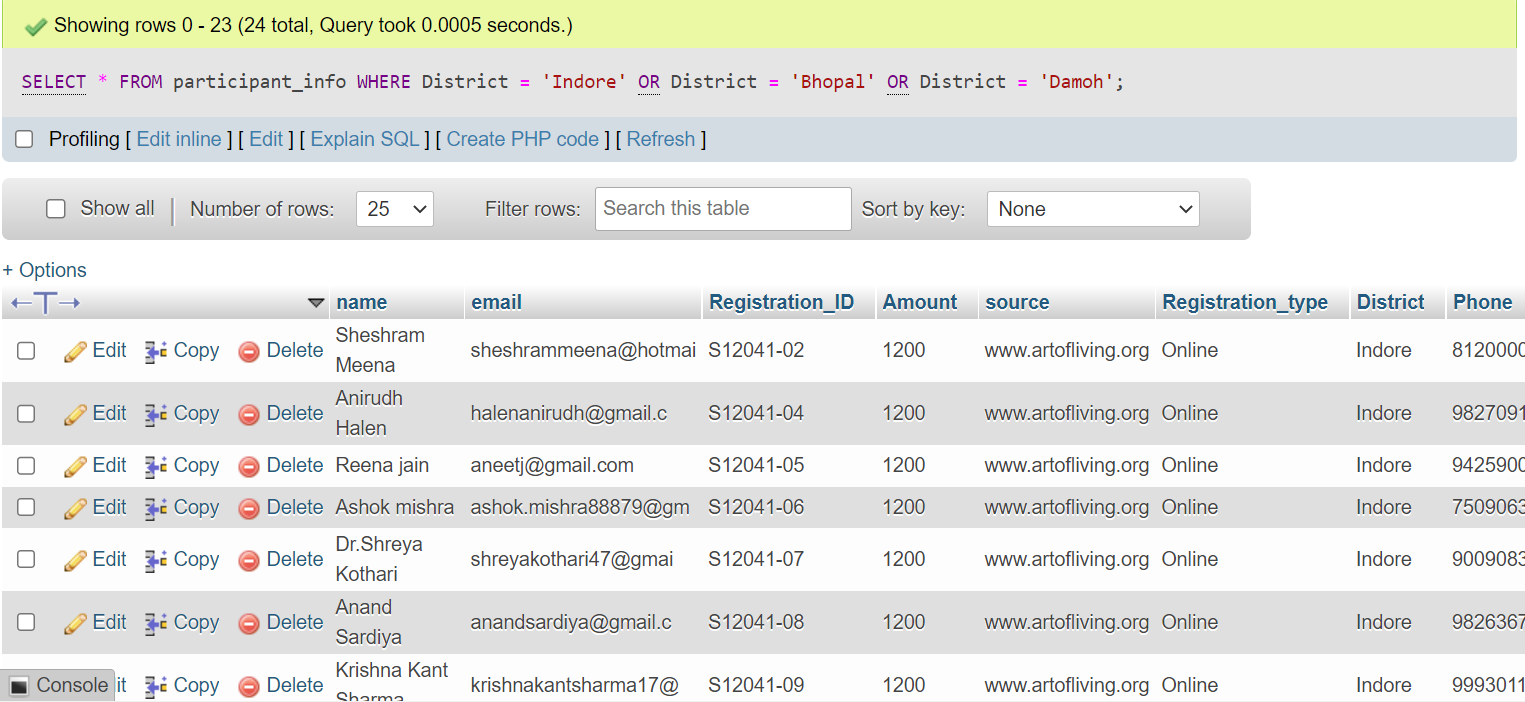
SELECT \* FROM participant\_info WHERE District = 'Indore' OR District = 'Bhopal' OR District = 'Damoh';

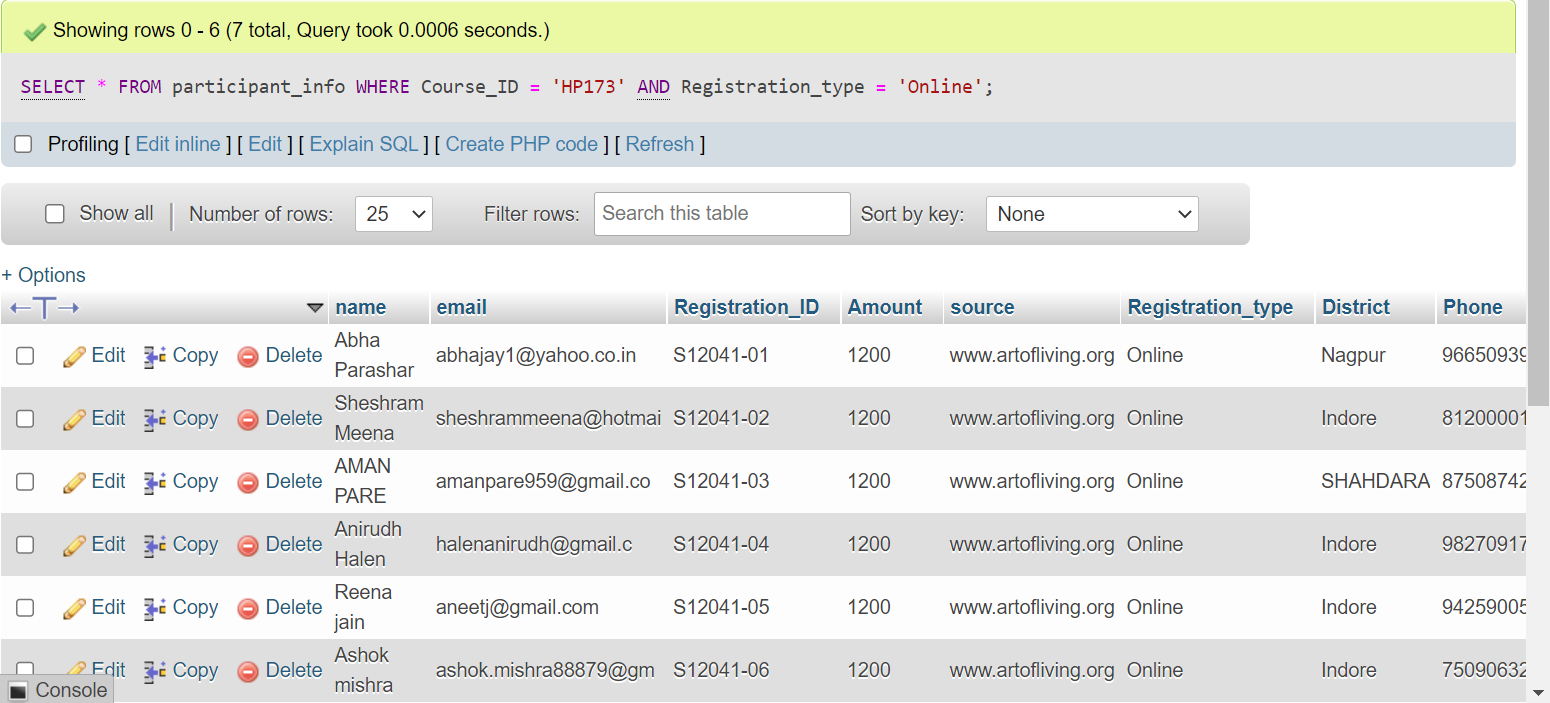
/\* Set operation to get list of participants whose Course\_ID is HP173 and Registration\_type is 'Online'\*/

SELECT \* FROM participant\_info WHERE Course\_ID = 'HP173' AND Registration\_type = 'Online';









# Functions and Procedures

Create a Function and Procedure. State the objective of the function / Procedure. Run and display the results.

--Functions

/\* Create function to check if Number\_of\_courses>4 in advance\_program\_details\*/

DELIMITER &&

CREATE FUNCTION check\_courses

(

    @Number\_of\_courses INT

)

RETURNS BIT

AS

BEGIN

    DECLARE @result BIT

    IF @Number\_of\_courses > 4

        SET @result = 1

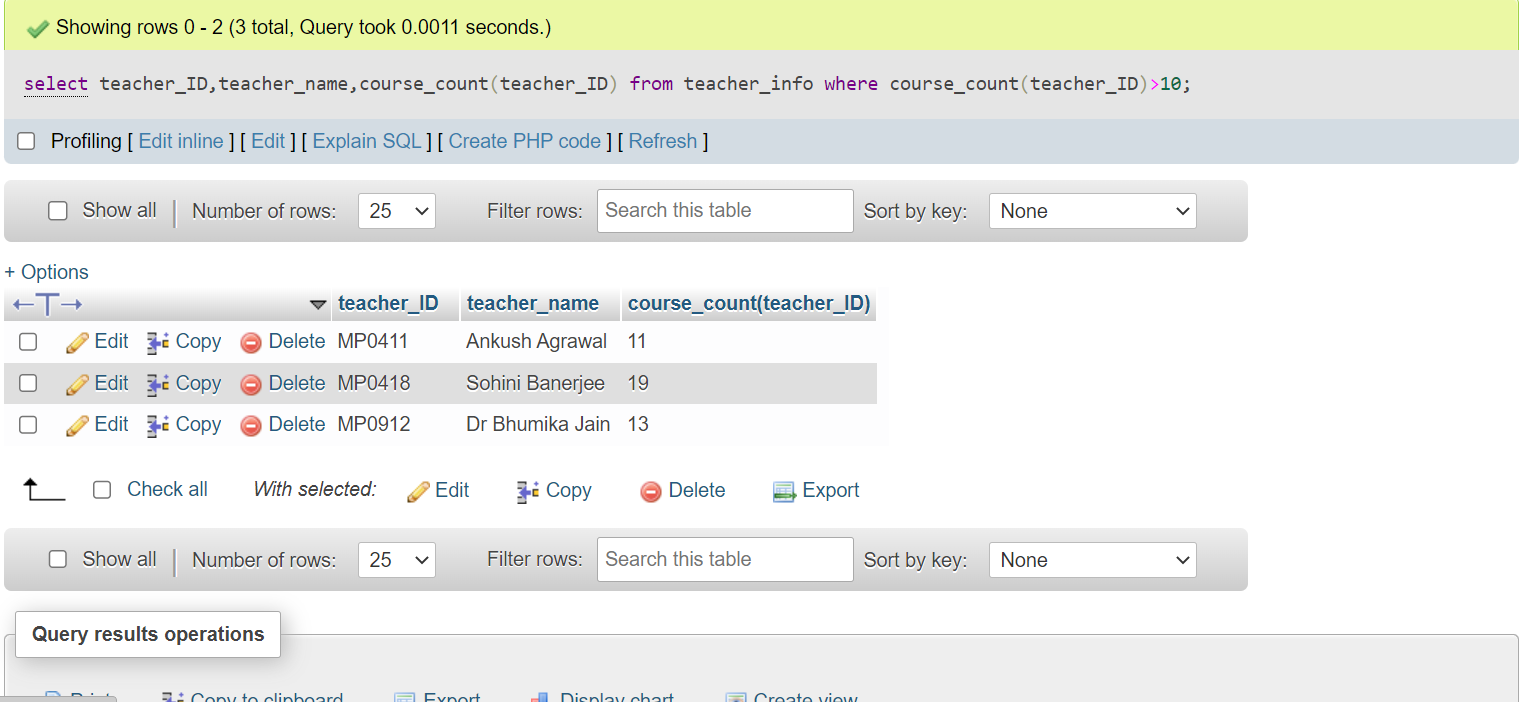
    ELSE

        SET @result = 0

    RETURN @result

END &&

DELIMITER ;



--Procedure--

/\* Create procedure to display rows where Number\_of\_courses>4 in advance\_program\_details\*/

DELIMITER &&

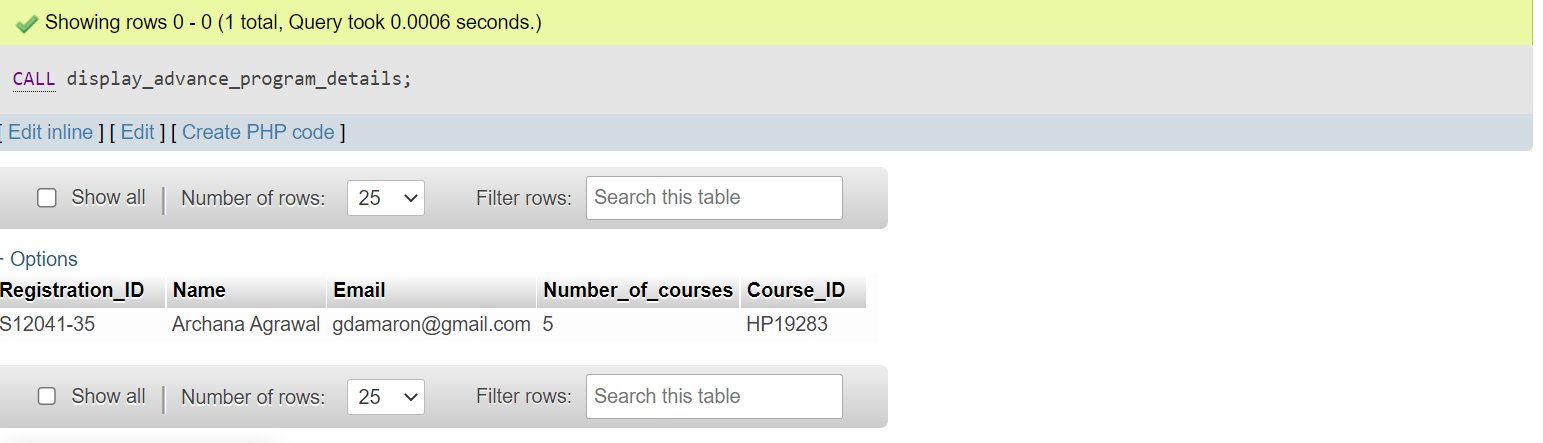
CREATE PROCEDURE display\_advance\_program\_details

BEGIN

SELECT \* FROM advance\_program\_details WHERE Number\_of\_courses>4;

END &&;

DELIMITER;



# Triggers and Cursors

Create a Trigger and a Cursor. State the objective. Run and display the results.

/\* Create trigger to increment no\_of\_courses by 1 in teacher\_info table if teacher\_ID in teacher\_info table is same as teacher\_ID in course\_details table\*/

CREATE TRIGGER increment\_no\_of\_courses

AFTER INSERT ON course\_details

FOR EACH ROW

BEGIN

UPDATE teacher\_info SET no\_of\_courses = no\_of\_courses + 1 WHERE teacher\_ID = NEW.teacher\_ID;

END;

/\*cursor to obtain the info of first and last row from participant\_info table\*/

DECLARE cursor1 CURSOR FOR SELECT \* FROM participant\_info WHERE Course\_ID = HPY FROM participant\_info);

# Developing a Frontend

The frontend should support

1. Addition, Modification and Deletion of records from any chosen table
2. There should be a window to accept and run any SQL statement and display the result

